INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2014; **74**:846–855
Published online 24 January 2014 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/fld.3878

A linear solver based on algebraic multigrid and defect correction for the solution of adjoint RANS equations

Malte Förster[1] and Anna Pal[2,*,†]

[1]*Fraunhofer Institute for Algorithms and Scientific Computing, Schloss Birlinghoven, 53754 Sankt Augustin, Germany*
[2]*German Aerospace Center, Institute of Aerodynamics and Flow Technology, Lilienthalplatz 7, 38108 Braunschweig, Germany*

SUMMARY

A new solution approach for discrete adjoint Reynolds-averaged Navier–Stokes equations is suggested. The approach is based on a combination of algebraic multigrid and defect correction. The efficient interplay of these two methods results in a fast and robust solution technique. Algebraic multigrid deals very well with unstructured grids, and defect correction completes it in such a way that a required second-order accuracy of the solution is achieved. The performance of the suggested solution approach is demonstrated on a number of representative benchmarks. Copyright © 2014 John Wiley & Sons, Ltd.

Received 15 April 2013; Revised 18 December 2013; Accepted 28 December 2013

KEY WORDS: linear solvers; Reynolds averaged Navier–Stokes; compressible flow; finite volume; optimization

1. INTRODUCTION

It is a common practical observation that linear systems originating from the discrete adjoint RANS equations are often very stiff. Various factors might contribute to the stiffness of these systems, among which are unstructured grids, unfavorable flow conditions, turbulence, low numerical dissipation of employed discretization scheme, boundary conditions, reference flow solution, and so on. Moreover, when problems of practical interest are solved, the obtained linear systems are often very large, being based on grids with several million points. Although these systems are considered sparse, a single unknown might still be coupled to up to tens or even hundreds of other unknowns. Matrices associated with these systems are as a rule strongly off-diagonally dominant and contain a big number of both positive and negative large off-diagonal entries. Such properties are usually undesirable by most iterative solution techniques.

As a result, efficient solution of these systems is a challenging task. Commonly used methods to solve such systems are, for example, geometric multigrid or various solvers based on incomplete lower-upper (ILU) factorization. In addition, stabilization with Krylov methods is needed for these linear solvers in most practical cases. However, even when combined with a Krylov solver, these mentioned solution approaches are not fully satisfactory, at least in the context of the DLR TAU code. Memory-efficient geometric multigrid often exhibits a poor convergence or stagnates, whereas ILU-based approaches become inapplicable to many large problems due to very large memory requirements of high-level ILU decompositions. Therefore, we were motivated to search for alternative solution approaches and eventually came to investigate the potential of algebraic multigrid (AMG).

*Correspondence to: Anna Pal, German Aerospace Center, Institute of Aerodynamics and Flow Technology, Lilienthalplatz 7, 38108 Braunschweig, Germany.
†E-mail: anna.pal@dlr.de

Algebraic multigrid is known to be a very efficient linear solver and even the optimal order method for certain classes of problems. However, linear systems that we deal with within this work violate several requirements imposed on the equations by AMG. Therefore, extensions of AMG are necessary to deal with these problems. Moreover, in order to obtain a working solution method, we do not apply a stand-alone AMG but combine it with defect correction. We started the work in this direction from considering the linear systems corresponding to the adjoint Euler equations, and the respective results were published in [1]. The present work is a continuation of this earlier work, which extends the method so that it can be applied to the adjoint RANS equations and, moreover, can be used in a parallel mode for large applications of a practical interest. As in [1], we rely in the current work on the SAMG library [2]. However, readjustment of several AMG components within this library (e.g., coarsening, smoothing) is necessary for an appropriate performance of the solver.

The paper is organized as follows. In Section 2, we describe the nonlinear flow equations and the employed discretization. Discrete adjoint equations are formulated in Section 3. A detailed description of the employed AMG approach as well as of the defect correction algorithm, the combination of which form the basis of the solution strategy, is presented in Section 4. Finally, numerical results are shown and discussed in Section 5.

2. FLOW EQUATIONS AND THEIR DISCRETIZATION

2.1. Governing flow equations

The governing flow equations, which are considered in this work are the compressible Reynolds-averaged Navier–Stokes (RANS) equations. As a one-equation turbulence equation, we consider either the original Spalart–Allmaras turbulence model [3] or the Spalart–Allmaras model with Edwards modifications [4].

In their conservative form in three dimensions, the RANS equations can be written in the following way

$$\nabla \cdot (f^c(W) - f^v(W)) = S(W), \tag{1}$$

where $W = (\rho, \rho u, \rho v, \rho w, \rho E, \rho\tilde{\nu})^T$ is the unknown conservative state vector, defined via the density ρ, the velocity vector $U = (u, v, w)^T$, the total energy E, and the turbulence variable in the Spalart–Allmaras model $\tilde{\nu}$. The vectors $f^c(W)$ and $f^v(W)$ represent the convective and viscous fluxes, respectively, and $S(W) = (0, 0, 0, 0, 0, s(W))^T$ is the source term, which is specific for the Spalart–Allmaras model.

2.2. Finite volume discretization

The governing equations are discretized via a finite volume method on unstructured grids of non-overlapping dual cells of a hybrid primary grid.

For the discretization of convective fluxes in the mean-flow equations, we employ the Jameson–Schmidt–Turkel scheme [5] with scalar dissipation, which is generalized for usage on unstructured grids. According to this scheme, the inviscid flux on face $\mathcal{F}_{ij}$ between two grid points $\mathcal{P}_i$ and $\mathcal{P}_j$ can be written as follows:

$$f^c_{ij} = \frac{1}{2}\left(f^c(W_i) + f^c(W_j)\right) \cdot n_{ij} - \frac{1}{2} D_{ij}, \tag{2}$$

where dissipation term D_{ij} is defined as

$$D_{ij} = \phi_{ij} \lambda^c_{ij} \left[\varepsilon^{(2)}_{ij}(W_j - W_i) - \varepsilon^{(4)}_{ij}\left(\nabla^2(W_j) - \nabla^2(W_i)\right)\right]. \tag{3}$$

In the aforementioned formula, ϕ_{ij} regulates the amount of dissipation on the face according to its size, λ^c_{ij} is the convective eigenvalue on this face, $\varepsilon^{(2)}_{ij}$ and $\varepsilon^{(4)}_{ij}$ control the levels of the two types of dissipation, and $\nabla^2(W_i)$ is the second undivided difference in point $\mathcal{P}_i$.

For the discretization of convective fluxes in the turbulence equation, an upwind scheme with piecewise-constant face reconstruction is employed.

DOI: 10.1002/fld

The viscous flux discretization in the mean-flow equations is carried out via the Green–Gauss formula, whereas the thin shear layer approximation is employed to discretize viscous fluxes in the turbulence equation.

For more details concerning the employed discretization scheme, we refer to the description of the discretization used in DLR TAU code [6].

3. DISCRETE ADJOINT EQUATIONS

3.1. The discrete adjoint Equations

Gradient-based optimization requires the computation of gradients of the cost function with respect to a set of specified parameters—the so-called design variables. When the number of these parameters is very large, a large number of gradients needs to be estimated. The adjoint method provides an efficient way to perform this—with the expense depending only weakly on the number of design variables.

In order to give a brief description of the adjoint method, we first write the finite volume discretization of the system (1) in the compact way

$$R(W, X, D) = 0. \tag{4}$$

In Equation (4), R is called the residual of the discretization, and it contains the discretization of all terms of the flow equation. D is a set of design variables, and $X = X(D)$ is the computational mesh.

Suppose we want to solve the following minimization problem

$$\begin{aligned} I(W, X, D) &\rightarrow \min_{D}, \\ R(W, X, D) &= 0, \\ T(X, D) &= 0, \end{aligned} \tag{5}$$

that is, we minimize the cost function $I(W, X, D)$ with respect to the set of design variables D, subject to two sets of constraints—the discrete RANS equations and the mesh deformation equations. The mesh deformation equations are defined via operator $T(X, D)$.

Via the adjoint-based approach, the gradient dI/dD is computed with the formula as follows:

$$\frac{dI}{dD} = \frac{\partial I}{\partial D} + \Lambda^T \frac{\partial R}{\partial D} + \hat{\Lambda}^T \frac{\partial T}{\partial D}. \tag{6}$$

In Equation (6), Λ and $\hat{\Lambda}$ are flow-adjoint and mesh-adjoint variables, which are obtained from solving the flow-adjoint and mesh-adjoint equations, respectively. In this work, we are solely concerned with the solution of the flow-adjoint equations, therefore, we skip the formulation of the mesh-adjoint ones. The flow-adjoint equations are written as follows:

$$\left(\frac{\partial R}{\partial W}\right)^T \Lambda = -\left(\frac{\partial I}{\partial W}\right)^T \quad . \tag{7}$$

A big advantage of the adjoint-based approach is that in order to evaluate the gradient of I with respect to any number of design variables, Equation (7) must be solved only once. After that, in order to evaluate derivatives of I with respect to these variables, the solution vector Λ is repeatedly substituted into formula (6), combined with the vectors corresponding to each design variable, dR/dD and dT/dD.

4. SOLUTION STRATEGY

System (7) is a large sparse system of linear algebraic equations. In this work, we develop a strategy for its solution based on a combination of AMG and defect correction. In the following, we present a brief description of these two methods and show how we combine them to obtain our solution approach.

DOI: 10.1002/fld

4.1. Algebraic multigrid

Unlike geometric multigrid, AMG does not directly employ any geometric information of a problem. All its components are constructed purely algebraically, relying entirely on the entries of a system matrix. Basic principles of AMG were first introduced by Brandt, McCormick, Ruge and Stüben [7–10].

An interested reader should refer to these monographs for a detailed description of the AMG method, which is not the scope of the present work.

The classical AMG algorithm consists of two phases—the setup phase and the solution phase. In the setup phase, all AMG components—the coarse levels, intergrid transfer operators, and coarse grid operators—are constructed. In the solution phase, these components are used in a multigrid cycling algorithm.

Construction of coarse levels and interpolation operators in the setup phase of the classical AMG method assumes M-matrices, which typically arise in the discretizations of elliptic scalar PDEs. For symmetric positive definite M-matrices, classical AMG is known to be a very robust and efficient solver. However, in this work, we are dealing with equations that originate from non-elliptic systems of PDEs and violate many requirements imposed by classical AMG. Resulting matrices are non-symmetric and usually far from being M-matrices. Therefore, extensions of classical AMG are required in order to solve these problems.

In the following, we describe on which AMG components—coarsening strategy, interpolation/restriction pattern, and smoothing algorithm—we base our solution approach.

4.1.1. Coarsening. In order to deal with a discretized system of PDEs, we employ the so-called point-based AMG strategy, that is, the same coarse grid hierarchy is used for every physical unknown. This strategy is combined with an aggregation type AMG approach, so that a single set of aggregates is built for all physical unknowns. The construction of the aggregates is based on a scalar matrix with a point level connectivity pattern, the so-called primary matrix. From practical experience, we found that the aggregates based on the primary matrix corresponding to density–density couplings within the system result in a robust and efficient approach. Therefore, we rely on this coarsening strategy in our work. The usage of strategies based on other physical unknowns or on various blockwise interpretations did not prove to give a more efficient AMG approach.

In parallel, the coarsening is carried out independently for each subdomain, leading to different coarsenings for varying processor counts. However, the quality of this coarsening is only affected near the boundaries of the subdomains. In order to reduce the resulting negative effect on the convergence rates, we reduce the aggregation size at the boundaries at the cost of a slightly reduced coarsening rate.

4.1.2. Interpolation, restriction, and coarse grid operators. As it is typical for an aggregation-based AMG approach, piecewise constant interpolation is applied separately for every physical unknown within the constructed aggregates. On each level, the restriction operator is defined as the transpose of interpolation, and Galerkin operators are used as coarse-grid operators.

4.1.3. Smoother. Because standard AMG smoothers such as Gauss–Seidel and W-Jacobi turned out not to be robust enough for the problem under consideration, we focus on a more robust smoothing algorithm in form of ILU(0). In addition, a reordering of variables according to the Reverse–Cuthill–McKee algorithm (RCM) has proven to be very beneficial for the performance of ILU. In parallel computations, we perform ILU decomposition domain-wise and then merge the results given by each subdomain via the additive Schwarz method, which overlaps the subdomains at their boundaries by a given depth. In our approach, we use an overlap of depth 1.

4.2. Defect correction approach

The linear problem (7) originates from the second-order accurate finite volume discretization. Because of a low numerical dissipation in this discretization, this linear problem is stiff and has a strong off-diagonal dominance. Experience showed that a direct application of the derived AMG

DOI: 10.1002/fld

solver to this system was not efficient. In that case, difficulties were often faced already at the stage of coarsening. Even if the coarsening was successful, a fast divergence of the approach was commonly observed for large test cases.

The defect correction approach [11, 12] offers a strategy for solving second-order accurate problems. It relies on an auxiliary first-order accurate discretization of the same continuous problem. In this approach, a chosen iterative solver (in our case, AMG) is repeatedly applied within a sequence of first-order problems, converging, however, toward the solution of the target system.

The benefit of this method is due to the fact that first-order discretizations are usually considerably better conditioned and allow more efficient solution with iterative methods.

For convenience of presentation of defect correction algorithm, we rewrite the linear problem (7) in a compact way, following common notations of linear algebra

$$Ax = b, \tag{8}$$

where

$$A = \left(\frac{\partial R}{\partial W}\right)^T, \quad x = \Lambda, \quad \text{and} \quad b = -\left(\frac{\partial I}{\partial W}\right)^T \quad .$$

If A_1 is a first-order accurate linear operator, corresponding to the same underlying problem as A and $x^{(0)}$ is an initial approximation to the solution of (8), then the defect correction algorithm for problem (8) can be written as follows:

$$\begin{aligned} &\text{For } n = 1, 2, \ldots \text{ until convergence do} \\ &\qquad \text{Solve equation for correction:} \quad A_1 e^{(n)} = b - A x^{(n-1)}, \\ &\qquad \text{Update solution:} \quad x^{(n)} = x^{(n-1)} + e^{(n)}. \end{aligned} \tag{9}$$

Note that for increased stability, the defect correction loop itself is employed as a pre-conditioner within the generalized minimal residual (GMRES) method.

In order to complete the definition of our solution approach, we have to specify the following:

- Which particular first-order accurate operator is employed,
- Which iterative solver is used to solve the equation for correction,
- How precisely is the correction equation solved at each iteration.

The choice of these components is described in the following paragraphs.

First-order accurate operator. As a first-order accurate operator A_1, we employ the Jacobian of the finite volume scheme, which is defined as described in the following text. For discretization of the inviscid flux, we use the Jameson–Schmidt–Turkel scheme with a simplified dissipation operator D. For this, we cut down the dissipation formula (3) in the following way:

$$D^1_{ij} = \lambda^c_{ij} \varepsilon^{(2)}_{ij} (W_j - W_i), \tag{10}$$

where the coefficient $\varepsilon^{(2)}_{ij}$ is further simplified so that it only depends on two direct neighbors of face $\mathcal{F}_{ij}$. In its turn, the viscous flux discretization is not based on Gauss formula as was the original scheme but on thin shear layer approximation. The discretization of the turbulence equation is kept identical to the original scheme, because it already relies on first-order approximation only.

These simplifications have an important impact on the Jacobian operator, making it not only better conditioned but also considerably more sparse. It significantly reduces the memory needed for its storage and makes the application of an iterative solver faster and more efficient.

Iterative solver applied to correction equations. Because a large number of defect equations (9) have to be solved one after another within a defect correction loop, the efficiency of the employed iterative solver for their solution is crucial to the performance of the algorithm. In this work, we employ the AMG solver in the form described in Subsection 4.1. Practical experience shows that although this solver could not handle the linear systems based on the second-order accurate discretization, it is very efficient when applied to the systems based on the auxiliary first-order one.

DOI: 10.1002/fld

How precisely to solve correction equations. Because one correction equation has to be solved at each iteration of defect correction, it would be very expensive to solve these equations very precisely. In that case, the efficiency of the whole approach would be lost. Fortunately, practical experience shows that it is sufficient to solve these defect equations rather approximately. In our algorithm, we employ only one step of AMG for solution of defect equation at each iteration of algorithm (9). This makes each iteration rather cheap and minimizes the total CPU run time of the solution method when compared with methods based on a bigger number of AMG steps.

5. RESULTS

5.1. Considered configurations

In this section, we demonstrate the performance of the suggested solver for four test cases, based on the following geometries: L1T2 high-lift airfoil, LANN wing, DLR-F12, and DPW-4 wing–body–tail configurations. All of these geometries are supplied with hybrid grids, which are typical for RANS computations within the DLR TAU code. First, we demonstrate convergence of the solver for the adjoint problems formulated for these test cases. Further, we study parallel scalability of the solution approach on the example of the LANN wing test case.

5.1.2. L1T2 airfoil. L1T2 is a three-element high-lift airfoil, which, in this work, is considered at the angle of attack (AoA) of 21.4°. The flow speed is kept rather low, corresponding to the Mach number of 0.22. A hybrid grid of about seven thousand points, which is shown on the left side of Figure 1, is employed for computations in this test case.

5.1.3. LANN wing. The LANN wing test case is supplied with a hybrid grid of about five million points (the grid is shown on the right side of Figure 1). A transonic flow regime is considered in this test case (respective Mach number is 0.82). The angle of attack is rather low for this test case, corresponding to 0.6 degrees.

5.1.4. DLR-F12 wing-body-tail configuration. The DLR-F12 configuration is considered at 0 degrees angle of attack, and it is subjected to a subsonic flow (respective Mach number is 0.21). The computational grid, which contains about 10 million points, can be seen on the left side of Figure 2.

5.1.5. DPW-4 wing-body-tail configuration. This test case is taken from the fourth AIAA CFD Drag Prediction Workshop. The computational grid employed contains about 12 million points. The grid is depicted in the right side of Figure 2. A transonic flow regime is considered in this test case, and the angle of attack is 2.3°.

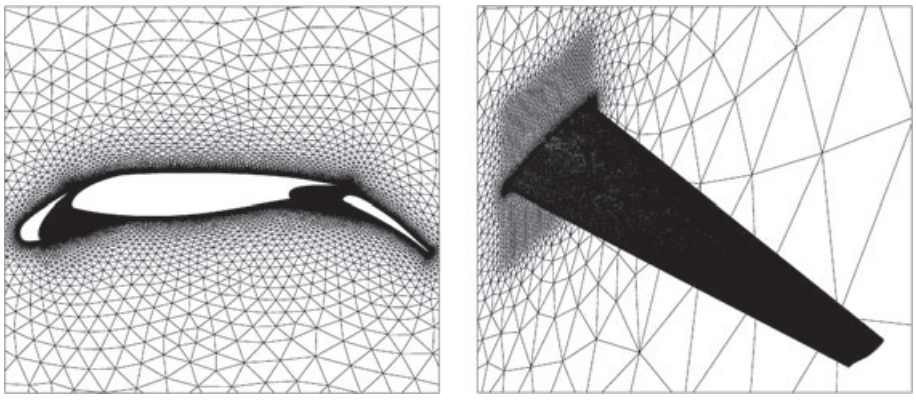

Figure 1. Grids for L1T2 airfoil (left) and for LANN wing (right).

DOI: 10.1002/fld

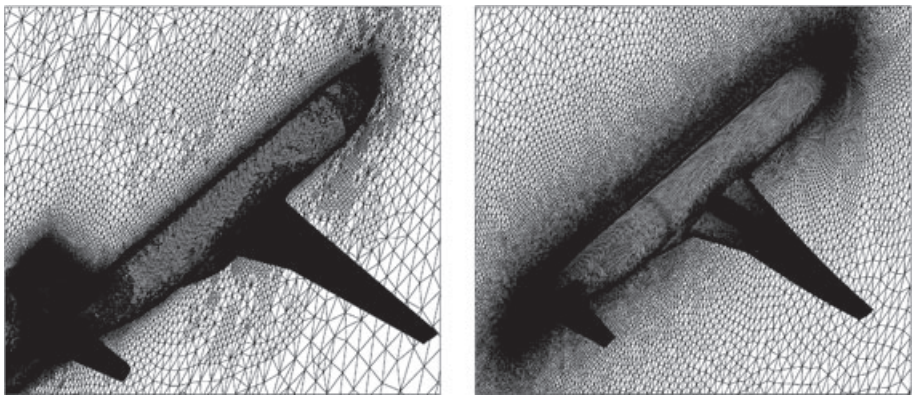

Figure 2. Grids for DLR-F12 (left) and for DPW-4 (right) wing-body-tail configurations.

Table I. Summarized information about the considered test cases.

Configuration	Number of grid points	Mach number	Reynolds number	AoA	Turbulence model	Cost function
L1T2	7×10^4	0.22	4×10^6	21.4	SAO	Drag
LANN	5×10^6	0.82	5.4×10^6	0.6	SAE	Drag
DLR-F12	10×10^6	0.21	1.3×10^6	0.0	SAO	Lift
DPW-4	12×10^6	0.85	5×10^6	2.3	SAO	Drag

SAO, Spalart–Allmaras turbulence model; SAE, Spalart–Allmaras model with Edwards modifications.

Table II. Employed number of processes and approximate CPU run times for the considered test cases.

Configuration	Number of processes	CPU time
L1T2	4	8 min
LANN	64	26 min
DLR-F12	112	2.5 h
DPW-4	120	1.7 h

The information about the considered test cases is summarized in Table I. From this table, one can also see that turbulent flow regimes were considered in all the test cases, and that a one-equation turbulence model (either Spalart–Allmaras turbulence model or Spalart–Allmaras model with Edwards modifications) was employed to model turbulence. As a cost function, either lift or drag coefficient was used.

5.2. Convergence of the solver

In the numerical experiments presented in this subsection, we demonstrate the convergence of the solver for four test cases described earlier. The stopping criterion is based on the reduction of the L2-norm of the total residual vector, that is, computed over all points and over all equations, by 10 orders of magnitude. This type of residual norm is minimized by GMRES method, within which the outer iterations of our solver are performed.

The computations were performed in parallel for all of the considered test cases. The numbers of employed MPI processes as well as the corresponding approximate CPU run times are summarized in Table II. Convergence curves for the test cases are shown in Figures 3 and 4, where the residual norm is shown versus iterations of GMRES. We remind that within the solution approach, each iteration of GMRES is preconditioned by one step of defect correction, which in its turn contains one cycle of AMG. A Krylov subspace of the size of 100 was used within GMRES method for all

DOI: 10.1002/fld

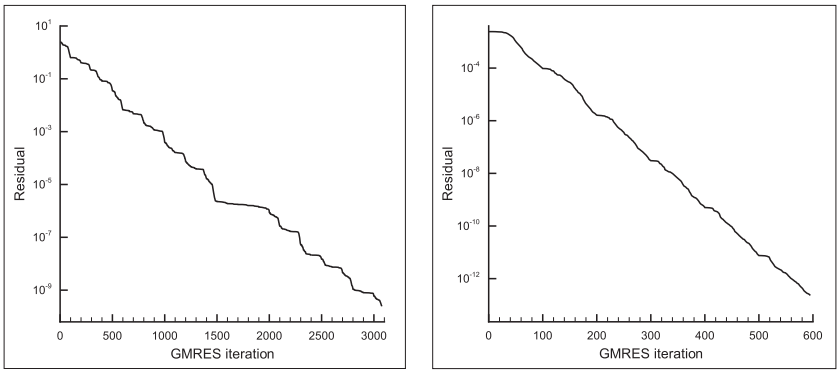

Figure 3. Convergence of the solver (norm of the total residual versus number of GMRES iterations) for the L1T2 configuration (left) and LANN wing (right).

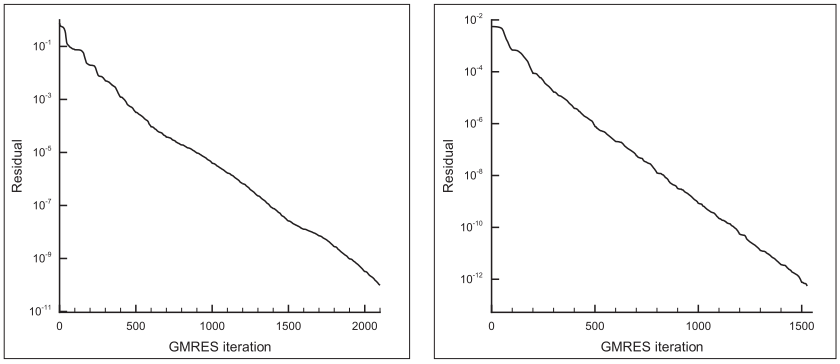

Figure 4. Convergence of the solver (norm of the total residual vs. the number of GMRES iterations) for the DLR-F12 configuration (left) and DPW-4 configuration (right).

the computations. The effect of GMRES restarts, performed after each 100 iterations, can be clearly seen in the slope of the convergence curve of the LANN wing test case (Figure 3, right).

As one can see from the convergence plots, all the test cases converged up to the prescribed residual threshold. In fact, the threshold value of 10^{-10} is untypically small for the considered class of problems. However, we wanted to demonstrate that the solver does not stagnate too early and, moreover, that its convergence does not deteriorate significantly as the residual is getting smaller. The convergence curves demonstrate that neither stagnation nor deterioration of convergence takes place.

One can also observe that in spite of the fact that the L1T2 test case is the only presented two-dimensional test case supplied with a rather coarse grid, the corresponding adjoint problem is very stiff and requires the largest number of iterations to achieve the prescribed residual threshold.

5.3. Parallel scalability

In this subsection, we investigate the scalability of the solution approach. As described in Section 4.1, several AMG components have to be adapted in order to work in combination with domain-based parallelism. In particular, the aggregative coarsening and ILU smoothing algorithms are modified at the processor boundaries. This leads to an increase of the overall workload when the problem is split into more and more subdomains. On the other side, this is needed in order keep the convergence of the solver stable with respect to the degree of parallelism. In other words, we are facing a trade off between numerical and parallel efficiency.

The plots in Figure 5 show the scaling of the solver applied to the LANN test case. For this, the test case is solved with increasing numbers of processes from 16 up to 128. The computations were

DOI: 10.1002/fld

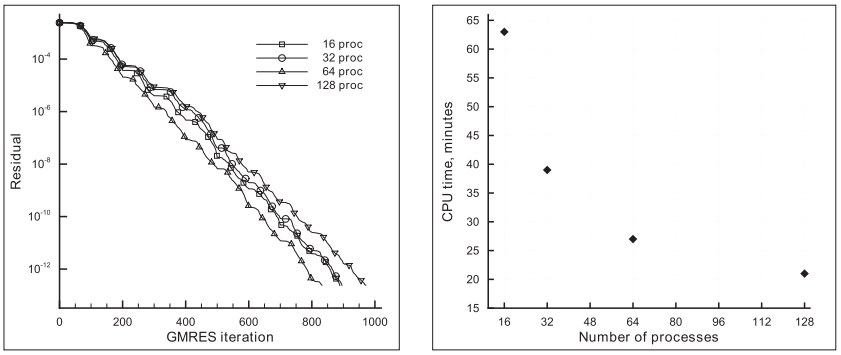

Figure 5. Scalability of the LANN test case with respect to convergence (left) and runtime (right).

performed on 16 nodes of a computational cluster, using from one to up to eight processes per node so that the required process counts (namely, 16, 32, 64, and 128) were obtained. The employed stopping criterion is the same as in the test cases earlier and corresponds to 10 orders of magnitude reduction in the norm of the total residual.

In the left graph of Figure 5, we monitor the convergence of the parallel algorithm for the performed computations in terms of iteration counts. As one can see from this plot, the number of iterations needed to reach the convergence criteria does not vary that much between computations performed with different number of processes. Furthermore, there is seemingly no straightforward dependency between convergence rate and degree of parallelism—the iteration number goes either slightly up or down as the number of employed processes varies.

In the graph on the right of Figure 5, we show the overall run times of the parallel computations against the number of processes employed. Because of the special treatment of processor boundary points, the complexity of the entire algorithm is not linear any longer. In addition, the sparse matrix times vector operations of the linear solver are memory bound way before CPU cores are fully utilized. Therefore, we do not expect linear scaling of CPU time in this figure. Nonetheless, a significant speedup in CPU time is achieved even in the computation performed on 128 processes as compared with the one performed on 64, going from four to eight processes per node, respectively.

6. CONCLUSIONS

A new approach for the solution of discrete adjoint RANS equations was presented. AMG and defect correction were combined together in order to provide the method that deals efficiently with linear systems coming from second-order accurate discretizations. The components of the AMG method had to be chosen and adjusted very carefully so that AMG could be applied in the context of the method. Moreover, certain readjustments of its components were carried out to assure that the method performs well in a parallel environment. The performance of the suggested solution approach was demonstrated on a number of representative benchmarks. It was shown that for all presented test cases, the method converges rather fast to a very low residual threshold. Also, no significant deterioration of residual reduction rate took place during convergence. Moreover, it was demonstrated that the parallel scalability of the approach is satisfying, opening up the possibility of approaching much larger 3D meshes on current hardware.

REFERENCES

1. Naumovich A, Förster M, Dwight R. Algebraic multigrid within defect correction for the linearized Euler equations. *Numerical Linear Algebra with Applications* 2010; **17**(2-3):307–324.
2. *SAMG package, Fraunhofer Institute for Algorithms and Scientific Computing*. (Available from: http://www.scai.fraunhofer.de/samg) [Accessed on 2 January 2013].
3. Spalart P, Allmaras S. A one-equation turbulence model for aerodynamic flows. *La Recherche Aérospatiale* 1994; **1**(1):5–21.

DOI: 10.1002/fld

4. Edwards J, Chandra S. Comparison of eddy viscosity-transport turbulence models for three-dimensional, shock-separated flowfields. *AIAA Journal* 1996; **34**(4):756–763.
5. Jameson A, Schmidt W, Turkel E. Numerical solutions of the Euler equations by finite volume methods using Runge–Kutta time-stepping schemes. *AIAA Paper 81-1259* 1981.
6. Gerhold T, Friedrich O, Evans J, Galle M. Calculation of complex three-dimensional configurations employing the DLR-TAU-code. *AIAA Paper 97-0167* 1997.
7. Brandt A. Algebraic multigrid theory: the symmetric case. *Applied Mathematics and Computation* 1986; **19**(1):23–56.
8. Brandt A, MacCormick S, Ruge J. *Algebraic Multigrid (AMG) for Automatic Multigrid Solution with Application to Geodetic Computations*. Institute for Computational Studies: POB 1852, Fort Collins, CO, 1982.
9. Ruge J, Stüben K. Algebraic multigrid. *Multigrid Methods* 1987; **3**:73–130.
10. Stüben K. Algebraic multigrid (AMG): experiences and comparisons. *Applied Mathematics and Computation* 1983; **13**(3-4):419–451.
11. Böhmer K, Hemker P, Stetter H. The defect correction approach. *Computing Supplement* 1984; **5**:1–32.
12. Koren B. Multigrid and defect correction for the steady Navier–Stokes equations. *Journal of Computational Physics* 1990; **87**(1):25–46.

DOI: 10.1002/fld